A Metrics-Based Approach to Intrusion Detection System Evaluation for Distributed Real-Time Systems

Authors: G. A. Fink, B. L. Chappell, T. G. Turner, and K. F. O'Donoghue; Information Transfer Technology Group, Code B35, Naval Surface Warfare Center, Dahlgren Division

Abstract

This paper describes a set of metrics that will help administrators of distributed, real-time (clustered) computer facilities to select the best intrusion detection system for their facilities. The metrics herein are the subset of our general metric set that particularly impact real-time and distributed processing issues. We discuss related works in this field, the role of intrusion detection in information assurance, some basic classes of intrusion detection systems, a general architecture of network intrusion detection systems, and the scorecard metrics and their application to real-time and distributed processing systems. Finally we discuss the lessons we learned using a preliminary version of the metric scorecard to test three commercial intrusion detection systems and the opportunities for further work in this area.

Keywords: *Intrusion Detection, Metrics, Evaluation, Real-Time, Distributed, Security.*

1 Introduction

Processing and cost requirements are driving future naval combat platforms to use distributed, real-time systems of commercial computers and networking components. The shift from specialized components to interoperable commercial hardware will increase the difficulty of meeting information security requirements. The move to distributed engineering and computing, increased automation, and reduced manning will strain the security paradigms used in current systems. Numerous subsystems will have to cooperate securely in an environment where milliseconds can make a life or death difference.

However, information security services often conflict with the performance requirements of highly distributed, real-time systems. As these systems grow more complex, the timing requirements do not diminish; indeed, they may become more constrained. Changing requirements imply re-engineering and continual re-evaluation of system security. Unfortunately, the threat to information systems is continually evolving as well and requires vigilant survey of available technologies to find those that best fit the information assurance needs of each system. Changes can always be made, but knowing whether or not a change will be an improvement is a challenge [1]. The idea of security itself has changed in this environment. Security used to be all about keeping "bad guys" out. But the rising field of information assurance (IA) considers security as an enabling technology. IA emphasizes that messages must be delivered without being intercepted or compromised and that authorized users must have access while others are denied. The paradigm shift causes re-evaluation of security engineering in positive as well as negative terms.

This paper focuses on an IA technology that is currently popular in the commercial sector: *Intrusion Detection* (ID). Like digital signatures and tamper-evident seals, ID assures integrity by making it clear when access controls have failed. In this paper, we describe a testing methodology we developed to evaluate ID products against a user-definable, dynamically-changing standard. We review the lessons learned during the development of this methodology and present an overview of the metric set we used in a test evaluation of three commercial products. The key distinctive of our approach is that we do not compare ID Systems (IDSs) against each other, but against a standard derived from mapping formalized user requirements to a standard set of metrics. Using a standard as the basis for comparison gives us scientific repeatability, and, because the evaluation is against a static set of metrics, the evaluation may be reused with the metrics given different weighting according to the needs of the next customer. Distributed, real-time, weapons-control systems like those we support have unique requirements that are seldom considered by market comparisons. This generalized approach will allow systems with such requirements to tailor evaluation of ID technologies to their specific needs.

Development of metrics implies some underlying formal taxonomy of IDSs, but to present one here would be beyond the scope of this paper. Although some work has been done in this area [2, 3, 4], nothing definitive has been accepted by the community at this time. Therefore, we have used information from these taxonomies combined with contrasting feature sets of existing IDSs and desired features from our user community as sources for our metrics. Surveys of IDSs too numerous to list have been presented, but reference [5] was particularly useful in defining terms and for a basic set of metrics for evaluating candidate IDSs (see [5], Appendix F). This work was seminal to our project.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	4/1/2002	Research Paper 4/1/2002

4. TITLE AND SUBTITLE
A Metrics-Based Approach to Intrusion Detection System Evaluation for Distributed Real-Time Systems

5. FUNDING NUMBERS

6. AUTHOR(S)
Fink, G.A.; Chappell, B.L.; Turner, T.G.; O'Donoghue, K.F.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Surface Warfare Center
Dahlgren, VA

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE
A

13. ABSTRACT ***(Maximum 200 Words)***

This paper describes a set of metrics that will help administrators of distributed, real-time (clustered) comuter facilities to select the best intrusion detection system for their facilities. The metrics herein are the subset of our general metric set that particularly impact real-time and distributed processing issues. We discuss related worksin this field, the role of intrusion detection in information assurance, some basic classes of intrusion detection systems, a general architecture of network intrusion detection systems, and the scorecard metrics and their application to real-time and distributed processing systems. Finally, we discuss the lessons we learned using a preliminary version of the metric scorecard to test three commercial intrusion detection systems and the opportunities for furtherwork in this area.

14. SUBJECT TERMS
IATAC Collection, intrusion detection, metrics, evaluation, real-time, distributed, securityu

15. NUMBER OF PAGES
8

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

We relied heavily on suggestions and tips from the source material for the recently published reference [6] in the design of our metrics and testbed. References [7] and [8] document extensive work done evaluating IDSs in DARPA's annual testing events. In the authors of these studies describe their testbed, LARIAT, and suggest some metrics and scoring for IDSs, but they do not present a well-define metric set or a way to transform customer requirements to their metrics. The NSS Group has published results of their IDS tests, but while they have standardized their tests, they have not presented standard metrics (see [9], pp. 164-169). We believe that our work complements that of others in the field and we have found no other formal, metrics-based evaluation methodology in the literature at this writing.

2 Intrusion Detection Systems

Even the best packet-filtering, stateful-inspection, proxy firewalls can miss many intrusions. By design, firewalls are boundary devices that are oblivious to the internal behavior of the network, systems, or users they monitor. ID provides defense-in-depth by double-checking the effectiveness of other access controls.

The purpose of an IDS is to detect both external attacks on, and internal misuse of computer and network resources, or information residing in these resources. Intrusions are most often thought of as originating from outside a trusted network. Unauthorized access from without may be achieved by traversing a leaky firewall, exploiting a security flaw, tunneling in through "benign" protocols, or entirely subverting security measures through an unprotected link to an external system. Threats from malicious "insiders" can materialize from compromised physical security, compromised passwords (masquerade), or users attempting to access information for which they do not hold the proper credentials. Both successful (where the attacker gained what he wanted from the system) and attempted attacks are of interest. Successful attacks may compromise the integrity, confidentiality, or availability of the resources or information. Attempted attacks serve as a warning of the threat level and of what kinds of resources are threatened.

ID technologies address an inherently (soft) real-time problem. Data must be analyzed as it arrives, analysis must be rapid and complete, and alerts must be issued in a timely manner to prevent further damage from intrusions. Additionally, IDSs that protect distributed, real-time systems must take into account the needs of those systems. They must not require significant resource overhead or introduce bottlenecks in the computing or network resources. They must execute deterministically and fail in a mode that does not hamper system performance.

2.1 Classification of an IDS

An IDS may be categorized by its detection mechanism: anomaly-based, signature-based, or hybrid. An anomaly-based IDS attempts to detect behavior that is inconsistent with "normal" behavior. Thus, these systems are sometimes called behavior-based. If an anomaly-based IDS detected hundreds of login attempts within a few seconds, it might generate an alert of suspicious activity. A signature-based IDS attempts to detect patterns in network traffic that are characteristic of known attacks. The terms "knowledge-based" and "misuse-based" are synonyms for "signature-based." This is a similar concept to anti-virus software on a PC that scans files and memory for known patterns of a computer virus. A hybrid IDS uses both technologies either in series or in parallel.

A signature-based IDS has the disadvantage that it will only detect previously known attacks, while an anomaly-based IDS may be able to detect new attacks. Distinguishing between "normal" and "anomalous" behavior, however, is the subject of much research. A constrained application environment may help constrain the definition of normal behavior making anomaly-based systems more appropriate. This maxim may apply to distributed, real-time systems such as those used for cluster super-computing where the network and protocols are tuned for highest performance. Anomaly-based systems are uncommon in commercial products where signature-based techniques dominate. However, many of the research endeavors have implemented a hybrid design.

An IDS may be further characterized as to its monitoring scope. An IDS may monitor one or more hosts, a network, or both. An IDS that monitors a host typically examines information available on the host such as log files. A multi-host IDS consists of cooperating host-based sensors that report raw or refined data to a remote analysis engine. An IDS that monitors a network collects and analyzes packets from the network. The types of resources consumed by the IDS will be determined by this characteristic. Support of host-based IDSs must consume some of the monitored host's resources. Nominal event-logging support for host IDSs has been shown to consume three to five percent of the monitored host's resources. Logging compliant with Department of Defense C2-level (Controlled Access Protection) security requires as much as twenty percent of the host's processing power [3, 10]. Obviously this is a concern for real-time systems. Multi-host IDSs consume network bandwidth by transmitting logging information. Network-based IDSs are often stand-alone, single-purpose machines that consume no other resources, but they may be run on a host that is simultaneously used for production work. In this case, they may also consume a large portion of the host's resources. Host-based IDSs also face another inherently real-time problem: when the host they run on is under attack, they

must quickly notify someone and possibly migrate to another host before they are compromised or disabled.

2.2 General Architecture of a Network IDS

An IDS may be deployed using a centralized or distributed architecture. In a centralized architecture, the collection and analysis of data occurs on the same device. The device might even be in-line, becoming a potential bottleneck in the network throughput. In a distributed architecture, a load-balancing system might send portions of traffic to multiple sensors that in turn send the collected data to one or more analysis stations and management consoles. Figure 1 shows a generalized architecture of network IDSs.

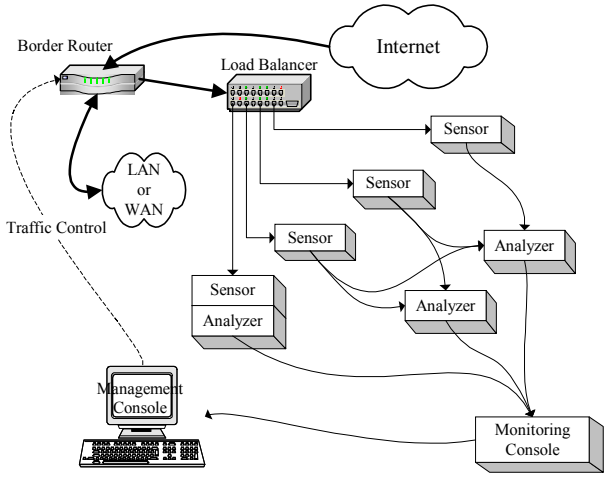


Figure 1: Generalized network IDS architecture

ID is a sequential process consisting of five subprocesses:

1. Load balancing: distributing all the network traffic among the available sensors
2. Sensing: separating suspicious from normal traffic
3. Analyzing: determining the nature and threat of suspicious traffic
4. Monitoring: allowing operator visibility into the threat, providing reports, and facilitating operator notification
5. Managing: allowing configuration of the IDS and control over incoming traffic in response to threat

These subprocesses may be engineered to overlap in many ways, but they are steps in an intrinsically sequential process. However, since there are numerous traffic flows and the relationship among these subprocesses is not one-to-one, many opportunities exist for optimally managing the throughput of the system via parallelism and distribution of computational load. Subprocesses one and five are optional, but subprocesses two through four are essential to network IDSs.

Figure 2 shows the cardinality of the relationships between the pairs of subprocesses in the overall sequential IDS process.

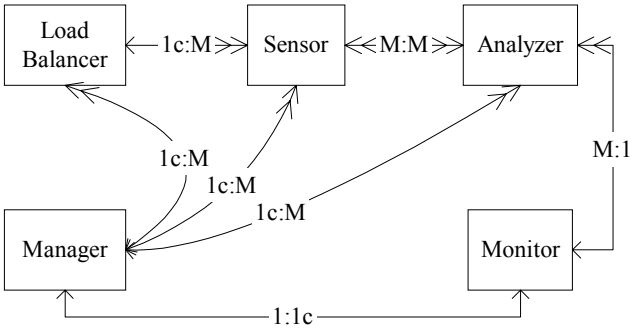


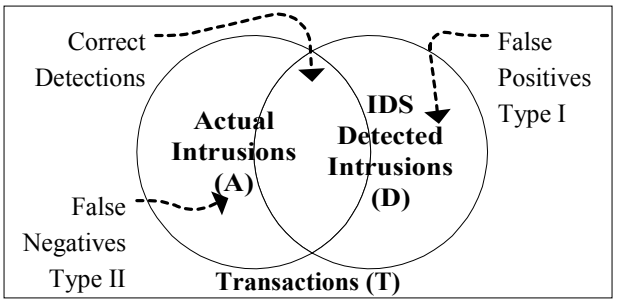
Figure 2: Relational cardinality of IDS subprocesses

Load balancing allows the IDS to efficiently utilize the processing power of the distributed sensors for scalability [11]. Load balancers may be in-line with a border router so that all traffic must go through it or all traffic may be mirrored to it. A one-to-many (1:M) relationship exists between load balancers and sensors. Load balancers may operate in a hierarchy but each sensor maps to exactly one load-balancer. This subprocess is optional, so the relationship is actually zero-or-one-to-many (1c:M, where the "c" indicates the conditional existence of the load balancer subprocess). Load balancers typically must be aware of Transmission Control Protocol (TCP) sessions so they can consistently send connection-oriented traffic to the appropriate sensor. If an IDS has no load-balancing component, the load may be statically spread out by placing sensors in separate subnets. Individual, statically placed sensors may overload or starve, and the protection of the network will be uneven. High-bandwidth load balancers may allow the IDS to collect traffic higher up in the network, closer to the border router. The result will be more efficient use of sensors and better protection for the monitored network. Load-balancing metrics are system throughput, cardinality of sensors supported, scalability of load balancers, and induced latency of traffic (because the load balancer is in-line or because traffic must be mirrored to it).

The sensors receive traffic from the load balancer (if any exists) and separate out the suspicious traffic for further analysis. Potentially a many-to-many (M:M) relationship between sensors and analyzers is possible. Frequently, sensory and analysis subprocesses are combined, reducing this relationship to one-to-one (1:1). Separating sensing from analysis may allow better throughput by offloading the analysis burden, but separation adds network overhead. Throughput may be improved more directly by efficient load balancing. Simple sensors are either signature-based or anomaly-based (see section 2.1). Sensor metrics are detection mechanism, degree of parallelism, and false positive and negative detection ratios.

Analyzers determine the threat level of the raw data collected by the sensors. Suspicious traffic may be either classified as an attack of some severity, or it may be dismissed as safe. Typically a many-to-one (M:1) relationship exists between the analyzers and the monitors they report to. Most IDSs that field multiple sensors separate analysis from monitoring. However, second-order analysis is often done by monitoring components. Analysis may be distributed according to scope (breadth of coverage) and depth (focus of analysis). Primary analysis determines threat severity. Secondary analysis determines scope, intent, or frequency of the threat. Accurate analysis may require storage of a significant amount of historical data that can be used to give context to a perceived threat. Good analysis can correlate one attack with another or determine that no such correlation is appropriate. Analysis metrics are scope of analysis (breadth), threat correlation capability (depth of analysis), and data storage requirements.

The monitoring subprocess presents a view of the threat to the operator. That view may be graphical or textual, and often provides some historical querying ability. Monitors are required to notify an operator whenever a threat is severe according to a security policy. Monitors must be tuned according to the traffic patterns of the protected network. Frequent alerts on trivial or normal events result in a high false-positive rate (Type I error) and lead to the IDS being ignored by the operators. On the other hand if the monitor is sufficiently desensitized, it will ignore true threats. False negatives (Type II error) will result, and the IDS will provide a misleading sense of security. See Figure 3 for an illustration of the meanings of false positive and false negative.



False Positive Ratio = |D - A| / |T|
False Negative Ratio = |A - D| / |T|

Figure 3: False Positive (Type I) and False Negative (Type II) Errors

In general, users should look for systems where the IDS's monitoring sensitivity can be adjusted so equality between false positive and false negative error rates can be achieved (see Figure 4). Of course the equal error rate is not always ideal. Given the choice, users might prefer to have lower Type II error at the expense of higher Type I error rates.

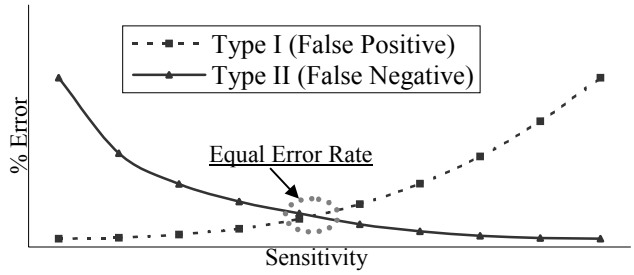


Figure 4: Error rate curves and Equal Error Rate

Monitoring components may provide data to a management console (if such a console exists) so that the detected threat may be dealt with. The relationship between a monitoring component and its optional management console is one-to-zero-or-one (1:1c). Metrics for the monitoring function are clarity of threat presentation, ease of policy maintenance, variety and interoperability of operator notification, and report generation capability.

Management consoles allow the operator to configure the IDS and to manage the threat by manipulating the incoming data stream via external devices like firewalls and routers. The management console is an optional feature of an IDS since sensor, firewall and router configuration may certainly be done in other ways, but the ability to automatically and accurately filter out offending traffic is key to a real-time response to threats. This is usually accomplished by a security policy that maps threats to automated actions. Policy must be accurate, for faulty policy risks shutting out legitimate users. The management console may also provide a user interface to configure the various sensors and other IDS subsystems. A 1c:M relationship exists between the management console and the various other components of the IDS. Without a management console, these components must be configured individually, but a console allows numbers of them to be configured centrally. Of course, decentralized management introduces network overhead. Metrics for the management function are ease of attack filter generation, accuracy and effectiveness of the filter, interoperability of the console with external devices, and component management effectiveness.

3 Evaluation Methodology

3.1 Scorecard Development

The centerpiece of our testing and evaluation methodology is a "scorecard" containing the set of general metrics and their definitions (see section 3.2). The metrics are general characteristics that we deemed relevant to any IDS. The metrics have been divided into three classes: Logistical (class 1), Architectural (class 2), and Performance (class 3). The two methods of observing each metric value were: analysis (direct observation in a

laboratory setting or source code analysis), and open source material (specifications, white papers or reviews provided by the vendor or users).

Each metric is designated to be measured by one or both of these methods. The key features of our testing methodology are:

1. Well-defined metrics
2. Discrete scoring
3. Flexible weighting

Well-defined metrics are observable, reproducible, quantifiable, and characteristic. By "characteristic" we mean that the metric must clearly differentiate between otherwise similar systems. Discrete scoring simplifies the process of assigning values to each metric for a given system. We chose to use scores with the discrete values zero through four, with higher scores interpreted as more favorable ratings. Our definition of each metric includes examples of low (0), average (2), and high (4) scores. Flexible weighting means that any consistent numeric system of weights can be used, discrete or continuous with upper or lower limits as defined by the scorer. The computation of weighted scores is specified in Figure 5. Using a larger range of weighting values will separate the field of products more distinctly. Negative weights may also be used to help distinguish where a feature is actually counterproductive.

$$S_j = \sum_{j=1,3} [\sum_{i=1,nj} (U_{ij} * W_{ij})]$$

where: **Sj** is the weighted overall score for metric class j
Uij is the unweighted score for metric i of class j
Wij is a real-valued weight of the ijth metric
nj is the number of metrics within class j
i is the index of the metrics within the jth class
j is the metrics class index (logistical = 1, etc)

Figure 5: Calculation of weighted scores

3.2 Scorecard Metrics

On the following pages we will discuss in greater detail the metrics we think are most applicable to IDSs in real-time environments. The metrics are grouped by class, followed by a representative metric from the class that includes examples of low, average, and high scores. For brevity's sake we have not included examples for each metric. The current complete scorecard is available from the authors. A prototype of this scorecard was used to evaluate three commercial IDS products: NFR Security's Network Intrusion Detection (NID) version 5.0 [12], Internet Security Systems' Real Secure version 5.0 [13], and Recourse Technology's Manhunt version 1.2 [14]. There was also an initial examination of an IDS from the research community, Autonomous Agents for Intrusion Detection (AAFID) [15].

Logistical Metrics. Logistical metrics measure the expense, maintainability, and manageability of an IDS. The metrics we have defined that are applicable to real-time in this area are shown in Table 1.

Distributed Management	Capability of managing and monitoring the IDS securely from multiple possibly remote systems.
Ease of Configuration	Difficulty in initially installing and subsequently configuring the IDS.
Ease of Policy Maintenance	The ease of creating, updating, and managing IDS detection and reaction policies.
License Management	The difficulty of obtaining, updating, and extending licenses for the IDS.
Outsourced Solution	The degree to which the IDS services are provided by an external entity.
Platform Requirements	System resources actually required to implement the IDS in the expected environment.

Table 1: Selected Logistical Metrics

Logistical metrics we have defined but not included in this paper are: *Quality of Documentation, Ease of Attack Filter Generation, Evaluation Copy Availability, Level of Administration, Product Lifetime, Quality of Technical. Support, Three Year Cost of Ownership, and Training Support*. A detailed example of the logistical metrics is *Distributed Management*:

- Low Score: Management of each node must be done at the node.
- Average Score: Nodes may be remotely managed, but either security, or degree of administrative control is limited.
- High Score: Complete management of all nodes may be done from any node or remotely. Appropriate encryption and authentication are employed.

We consider the administrative metrics *Ease of Configuration, Ease of Policy Maintenance,* and *License Management* applicable because products with low scores in these areas would be difficult to use in a distributed environment with multiple sensors. An *Outsourced Solution* might not be suitable for real-time or high performance environments if the agreement includes random vulnerability scanning. Such scans could disrupt system performance in a way that is not locally controllable. *Platform Requirements* give an indication of the system resources that will be consumed by the IDS in the resource-critical real-time environment.

Architectural Metrics. Architectural metrics are used to compare how well the intended scope and architecture of the IDS matches the deployment

architecture. The metrics we defined in this area are shown in Table 2.

Adjustable Sensitivity	Ability to change the sensitivity of the IDS to compensate for high false positive or false negative ratios.
Data Pool Selectability	Ability to define the source data to be analyzed for intrusions (by protocol, source and dest addresses, etc).
Data Storage	Average required amount of storage per megabyte of source data.
Host-based	Proportion of IDS input from log files, audit trails and other host data.
Multi-sensor Support	Ability of an IDS to integrate management and input of multiple sensors or analyzers.
Network-based	Proportion of IDS input from packet analysis and other network data.
Scalable Load-balancing	Ability to partition traffic into independent, balanced sensor loads, and ability of the load-balancing subprocess to scale upwards and downwards.
System Throughput	Maximal data input rate that can be processed successfully by the IDS. Measured in packets per second for network-based IDSs and Mbps for host-based IDSs.

Table 2: Selected Architectural Metrics

Architectural metrics we have defined but not included in this paper are: *Anomaly Based, Autonomous Learning, Host/OS Security, Interoperability, Package Contents, Process Security, Signature Based,* and *Visibility*. An illustrative example of an architectural metric is *Scalable Load-balancing:*

- Low Score: No load balancing
- Average Score: Load balancing via static methods such as placement
- High Score: Intelligent, dynamic load balancing

The significance of each metric in Table 2 to distributed, real-time systems is as follows: *Adjustable Sensitivity* allows tuning the IDS to optimal performance for the real-time environment. *Data Pool Selectivity* would allow the IDS to consider only protocols outside those typically used within the distributed cluster. *Data Storage* is a predictor of network bandwidth used in a distributed IDS. *Host-based* indicates the proportion of a monitored host's resources that the IDS will use. *Multi-sensor* measures the ability of an IDS to monitor a truly distributed system. *Network-based* IDSs will consume network resources by being in-line or via port mirroring. *Scalable Load-balancing* indicates whether an IDS will be able to grow as the system grows. The *System Throughput* metric helps determine whether the IDS will become a constraint on the processing ability of a real-time system.

Performance Metrics. Performance metrics measure the ability of an IDS to do a particular job and to fit within the performance constraints of the monitored system. The metrics we defined in this area are shown in Table 3.

Analysis of Compromise	Ability to report the extent of damage and compromise due to intrusions.
Error Reporting and Recovery	Appropriateness of the behavior of the IDS under error/failure conditions.
Firewall Interaction	Ability to interact with a firewall. Perhaps to update a firewall's block list.
Induced Traffic Latency	Degree to which traffic is delayed by the IDS's presence or operation.
Maximal Throughput with Zero Loss	Observed level of traffic that results in a sustained average of zero lost packets or streams. Measured in packets/sec or # of simultaneous TCP streams.
Network Lethal Dose	Observed level of network or host traffic that results in a shutdown/malfunction of IDS. Measured in packets/sec or # of simultaneous TCP streams.
Observed False Negative Ratio	Ratio of actual attacks that are not detected to the total transactions.
Observed False Positive Ratio	Ratio of alarms raised that do not correspond to actual attacks to the total transactions.
Operational Performance Impact	Negative impact on the host processing capacity due to the operation of the IDS. Expressed as a percentage of processing power.
Router Interaction	Degree to which the IDS can interact with a router. Perhaps it might redirect attacker traffic to a honeypot.
SNMP Interaction	Ability of the IDS to send an SNMP trap to one or more network devices in response to a detected attack.
Timeliness	Average/maximal time between an intrusion's occurrence and its being reported.

Table 3: Selected Performance Metrics

Performance metrics we have defined but not included in this paper are: *Analysis of Intruder Intent, Clarity of Reports, Effectiveness of Generated Filters, Evidence Collection, Information Sharing, Notification:*

User Alerts, Program Interaction, Session Recording and Playback, Threat Correlation, and *Trend Analysis*. The significance of each metric in Table 3 to distributed, real-time systems is as follows: *Analysis of Compromise* allows an administrator to determine which of the distributed systems is compromised for safer resource allocation. *Error Recovery and Reporting* indicates what an IDS will do when it fails or is overloaded. *Firewall Interaction*, *Router Interaction*, and *SNMP Interaction* all help determine what means are available for a near real-time automated response to an intrusion. *Induced Traffic Latency* measures the impact an IDS will have on network throughput. *Maximal Throughput with Zero Loss* indicates how effective the IDS will be given the expected traffic flow in the network to be protected. *Network Lethal Dose* tells the bandwidth where the IDS will fail to operate correctly leaving the system unprotected. *Observed False Negative Ratio* and *Observed False Positive Ratio* measure the accuracy of the IDS and the degree that its coverage will need to be extended with other security measures. The remaining two metrics have obvious bearing on real-time systems. An illustrative example of performance metrics is *Error Reporting and Recovery:*

- Low Score: No notification, no log, no indication that an error has occurred. Fatal errors cause system to hang indefinitely.
- Average Score: Failure is logged and user is notified at some point in the future when the IDS is able. Fatal errors cause cold reboot of entire machine
- High Score: Failure is reported near real time via attack notification channels. Fatal errors cause restart of application(s) or service(s).

3.3 Deriving Weights from User Requirements

To provide meaningful results, the metrics need to be weighted carefully according to the intended environment where the IDS will function. The weighting of metrics is derived from an analysis of the requirements of the prospective IDS procurer. To effectively use the scorecard, the procurer must have clearly defined requirements. The actual form of the requirements is flexible. This section suggests one possible algorithm for mapping requirements to a scorecard weighting (see also Figure 6). The user first lists his IDS requirements in a partial ordering from least important to most. Requirements should be stated in positive form or converted to the positive where possible to reduce unnecessary negative weights. Next, the first requirement (least important) should be assigned the lowest weight (e.g., one). Other requirements may then be assigned increasing weights in proportion to their relative importance. Since the ordering of requirements is partial, it is acceptable to have duplicate weights. After the requirements are weighted, each metric is assigned a weight equal to the sum of the weights of the requirements it contributes to. Weighted scores for each IDS under evaluation are computed using the formula in Figure 5.

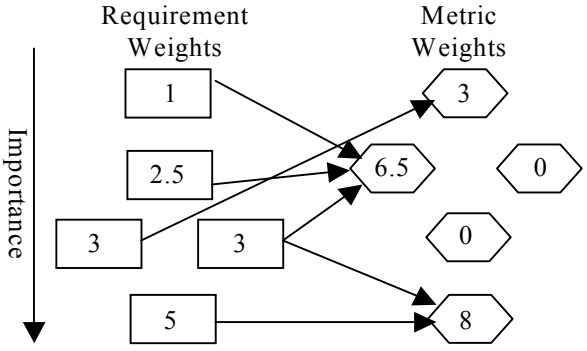


Figure 6: Requirement to Metric Weighting Example

Mapping of requirements to metric weights is an area where we hope to do more work in the future. A concrete mapping example will do much to mature both the scorecard and the mapping process. Mapping these requirements to numeric weights will always be somewhat subjective, but as long as the weighting accurately and consistently reflects the goals of the procurer's organization, the scorecard methodology will work effectively.

No security problem is purely technical, and much of determining the user's requirements has more to do with policy decisions than anything else. An organizational security policy that states the goals, acceptable uses, and constraints on the system in terms of security is critical. Without this organizational agreement, it will be impossible to determine what to monitor, when or whom to alert, or the degree of threat a potential intrusion presents [16].

The goal of the weighting strategy should be to reflect accurately the importance of each metric relative to the others for the system to be protected. For real-time systems, emphasis should be placed on speed and accuracy of attack recognition and on the ability of the IDS to automatically react via firewall, router, Simple Network Management Protocol (SNMP), etc. For distributed systems, care should be taken to consider the impact of trust among the component hosts. When one host is compromised, other systems that trust it may be very easily compromised in ways that may look like normal interactions between hosts. The result is an exploit that is difficult to detect and nearly impossible to root out. In this situation it is critical to catch the initial compromise of the first component host and isolate it. Distributed systems then, should put emphasis on reducing the false negative ratio to the lowest possible level accepting an increased false positive alert ratio in the process. Logging of historical traffic is also key to *ex post facto* unraveling the compromise of a complex distributed system.

4 Lessons Learned and Future Work

During the early stages of the evaluation, we learned several important lessons as we applied the testing methodology. First, to collect performance related metrics of an IDS, a simple flooding of the network being monitored with meaningless data is not sufficient [6]. Such a method is sufficient for determining the maximum throughput of a network device such as a switch or a router, but for an IDS, the data portion of an IP packet should have realistic content. The reason is that while some IDSs analyze only the header of the IP packet (e.g., source and destination address, port number, etc.), others also analyze the data portion of the packet. If packets with random data are used to generate background traffic, then the IDS that analyzes both the header information and message data will not be realistically tested. IDSs perform differently in the presence of different kinds of network traffic. Distributed systems with high levels of inter-host trust on a high-speed LAN will have distinctive traffic compared to that of a web server in an e-commerce shop. Commercial IDSs will often be geared toward the latter and not perform well in the former situation. The best way to evaluate any IDS is to use real traffic (live or recorded) from the site where the IDS is expected to be deployed [6]. Continual re-evaluation is especially important since vendors rapidly update their products.

A second lesson learned is that a few of the formal metrics we defined are very difficult (perhaps impossible) to observe. One such metric is the "observed false negative ratio." The user may never be aware of an instance of the IDS failing to detect an attack. To overcome this we replayed canned data with known attack content on the test network. However, even the definition of an attack is not always clear. What may be viewed as a single attack by one classification system may be legitimately seen as several attacks in another.

Potential future work includes weighting the score-card metrics with respect to the environment of a working distributed, real-time system; evaluating the current direction of IDS work in the research and development community; and a continued evolution of the metrics and testing methodology. The metrics and their definitions are best refined as lessons are learned while evaluating systems. We would like to expand the scorecard metrics to capture the human dimension of IDS as well. We hope to collaborate in the future with others who have fielded extensive IDS testbeds and adapt our metric set to their testing activities. Finally, we hope that our work will be of service to the growing population of IDS users and procurers.

5 References

[1] Brett Chappell, Glenn Fink, Karen O'Donoghue, David Marlow, *Intrusion Detection and Response for Real-time Distributed Naval Systems,* Parallel and Distributed Computing Practices – Special Issue: Security for Mission Critical Real-Time Systems, October, 2001.

[2] Stefan Axelsson, *Intrusion Detection Systems: A Survey and Taxonomy*, Technical report, Computer Engr Dept, Chalmers Univ of Technology, Goteborg, Sweden, 2000

[3] Hervé Debar, Marc Dacier, Andreas Wespi, *Towards a Taxonomy of Intrusion-Detection Systems*, Computer Networks, Vol 31, 1999, pp. 802-822.

[4] Lawrence R. Halme and R. Kenneth Bauer. *AINT misbehaving - a taxonomy of anti-intrusion techniques*. In Proceedings of the 18th National Information Systems Security Conference, pages 163-172, October 1995

[5] Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel, Ed Stoner, *State of the Practice of Intrusion Detection Technologies*, CMU/SEI-99-TR-028, SEI, Carn-egie Mellon Univ., Pittsburgh, PA 15213, Document available on the web at http://www.sei.cmu.edu/publications/ documents/99. reports/99tr028/99tr028abstract.html.

[6] Marcus Ranum, *Experiences Benchmarking Intrusion Detection Systems*, http://www.nfr.com/forum/white-papers/ Benchmarking-IDS-NFR.pdf, 2001.

[7] Rossey, Cunningham, Fried, Rabek, Lippmann, Haines, and Zissman, Lincoln Laboratory, Massachusetts Institute of Technology, *LARIAT: Lincoln Adaptable Real-time Information Assurance Testbed*, IEEEAC paper #033, 2001

[8] Haines, Rossey, Lippmann, and Cunning-ham, Lincoln Laboratory, Massachusetts Institute of Technology,

[9] Intrusion Detection Systems, Group Test (Edition 2), Published by the NSS Group, Oakwood House, Wenning-ton, Cambridgeshire, PE28 2LX, England, 2001. Document available on the web at http://www.nss.co.uk/ default.htm.

[10] The Department of Defense Trusted Computer System Evaluation Criteria, DOD-5200.28-STD, US DoD, 1985

[11] Top Layer Networks, Inc.,2400 Computer Drive, West-boro, MA 01581, 2002. Document available on the web at: http://www.toplayer.com/pdf/ iss_tln_GigResults.pdf.

[12] Network Flight Recorder, Inc., NFR NID, see product brochure at the company's website: http://www.nfr.com/ products/NID.

[13] Internet Security Systems, RealSecure, see product brochure at the company's website: http://www.iss.net/ securing_e-business/security_products/intrusion_detection/

[14] Recourse Technologies, Inc., ManHunt, see product brochure at the company's website: http://www.recourse.com/ products/manhunt/hunt.html

[15] E. H. Spafford, D. Zamboni, *Intrusion Detection using Autonomous Agents*, Computer Networks 34, 2000. See also http://www.cerias. purdue.edu/ homes/aafid/

[16] Rebecca Gurley Bace, Intrusion Detection, Macmillan, 2000, pp. 217-227.

For further information, or to obtain a copy of the IDS metric scorecard, please contact Glenn Fink at *finkga@nswc.navy.mil* or Brett Chappell at *chappellbl@nswc.navy.mil*.